INTERWOVEN

OpenDeploy® Reference

Release 5.5.1

INTERWOVEN

Interwoven, Inc.
803 11th Ave.
Sunnyvale, CA 94089
http://www.interwoven.com
Printed in the United States of America
Release 5.5.1
Part # 90-00-20-21-00-551-750

Table of Contents

Chapter 4: Command-Line Tools 75

About This Book

OpenDeploy Reference is a manual that contains reference material on OpenDeploy® configuration DTDs and command-line tools (CLTs). Use this information to find information on a specific DTD or CLT quickly and easily.

If you are using OpenDeploy in conjunction with TeamSite®, you should also know TeamSite functionality and terminology. Many of the operations described in this manual require *root* or *Administrator* access to the OpenDeploy host server. If you do not have root or Administrator access to the OpenDeploy host server, consult your system administrator.

This manual uses the term "Windows" to indicate any supported version of the Microsoft Windows operating system, such as Windows NT® or Windows® 2000.

This manual uses the term "UNIX" to indicate any supported flavor of the UNIX® operating system.

Windows: Users should be familiar with either IIS or Netscape® Web servers, and with basic Windows server operations such as adding users and modifying Access Control Lists (ACLs).

UNIX: Users of this manual should be familiar with basic UNIX commands and be able to use an editor such as emacs or vi.

It is also helpful to be familiar with regular expression syntax. If you are not familiar with regular expressions, consult a reference manual such as *Mastering Regular Expressions* by Jeffrey Friedl.

Notation Conventions

This manual uses the following notation conventions:

Convention	Definition and Usage
Bold	Text that appears in a GUI element (for example, a menu item, button, or element of a dialog box) and command names are shown in bold. For example: Click **Edit File** in the Button Bar.
Italic	Book titles appear in italics. Terms are italicized the first time they are introduced. Important information may be italicized for emphasis.
`Monospace`	Commands, command-line output, and file names are in monospace type. For example: The `iwodstart` command-line tool starts an OpenDeploy deployment task.
`Monospaced italic`	Monospaced italics are used for command-line variables.For example: `iwodstart` *`deployment`* This means that you must replace *`deployment`* with your values.
`Monospaced bold`	Monospaced bold represents information you enter in response to system prompts. The character that appears before a line of user input represents the command prompt, and should not be typed. For example: **`iwodstart`**
`Monospaced bold italic`	Monospaced bold italic text is used to indicate a variable in user input. For example: **`iwodstart`** ***`deployment`*** means that you must insert the values of *`deployment`* when you enter this command.
`[]`	Square brackets surrounding a command-line argument mean that the argument is optional.

Convention	Definition and Usage
|	Vertical bars separating command-line arguments mean that only one of the arguments can be used.

Other OpenDeploy Documentation

In addition to this Administration Guide, OpenDeploy includes the following documentation components:

- *OpenDeploy Administration Guide*
- *OpenDeploy Release Notes*
- Online help

OpenDeploy Administration Guide

OpenDeploy Administration Guide is a guide to install, configure, and use OpenDeploy ®. It is primarily intended for webmasters, system administrators, and those involved in deploying content between development servers and production servers.

OpenDeploy Release Notes

OpenDeploy Release Notes contains supplemental and late-breaking information regarding OpenDeploy not found in the other documentation. Refer to the OpenDeploy Release Notes for information regarding supported platforms, installation requirements, new features and enhancements, and known issues.

Online Help

OpenDeploy includes online help topics associated with each window in the OpenDeploy user interface. You can access the associated help topic by clicking the Help link in the upper right-hand corner of the window. The help topic will appear in a separate browser window that you can move and resize. You can display a navigation panel that provides you access to all the help topics by clicking the Show Navpane button. This panel provides you the ability to access help topics through the contents, index, and by using keyword searching.

Chapter 1

Deploy Server Configuration DTD

The deploy server configuration DTD specifies the XML rules for the base server configuration file on hosts with the base server software installed, and for the receiver configuration file on hosts with the receiver software installed. This file defines settings for server configurations, including what communications ports are being used to communicate with the other OpenDeploy software components. The base server and receiver files also contains default settings that apply if those settings are not defined in the deployment configuration file.

Deploy Server Configuration DTD

This section contains the XML code contained within the deploy server configuration DTD.

```
<!-- ============================================================ -->
<!--           The Deploy Server Configuration DTD                -->
<!-- ============================================================ -->

<!-- ============================================================
     Note: The following sub-elements are *required*:
           - allowedHosts
           - localNode
           - listenerProperties
           - schedulerProperties
     ============================================================ -->
<!ELEMENT deployServerConfiguration ( localNode,
                                      oldOdHome?,
                                      initiatorProperties?,
                                      listenerProperties,
                                      transportProperties?,
                                      teamsiteProperties?,
                                      schedulerProperties,
                                      allowedHosts,
                                      logRules?
                                    ) >
```

```
<!-- ============================================================
     The following properties define what incoming connections
     are allowed or denied and what port to bind on, and other
     miscellaneous listener/transport parameters.
     ============================================================ -->

<!-- ============================================================
     NOTE: The following specification is syntactically correct
           as far as XML is concerned, but not expressive enough
           for the purposes of OpenDeploy.
           Thus, please be aware that while all the ssl*
           attributes and keyFile attribute are optional - if you
           decide to use the ssl* attributes you MUST use ALL of
           them and CANNOT use the keyFile attribute.
           Conversely, if you decide to use the keyFile attribute,
           you CANNOT use ANY of the ssl* attributes.
     ============================================================ -->

<!ELEMENT localNode EMPTY >
<!ATTLIST localNode
      host             CDATA                       #REQUIRED
      sslCertificate   CDATA                       #IMPLIED
      sslPrivateKey    CDATA                       #IMPLIED
      sslCaCertificate CDATA                       #IMPLIED
      sslCiphers       CDATA                       #IMPLIED
      sslVerifyPeer    (require|request|none)      "none"
      keyFile          CDATA                       #IMPLIED
      >

<!-- ============================================================
     This element/attribute must be defined to enable
     communications with a [supported] older version of
     the OpenDeploy product.
     The 'path' attribute should define the base location of the
     previous version of OpenDeploy on *this* node.
     ============================================================ -->
<!ELEMENT oldOdHome EMPTY>
<!ATTLIST oldOdHome
      path               CDATA         #REQUIRED
      >

<!ELEMENT initiatorProperties EMPTY >
<!ATTLIST initiatorProperties
      name              (InterwovenOpenDeploy) #FIXED "InterwovenOpenDeploy"
```

```
      pendSessions       (yes|no)      "no"
      >

<!ELEMENT listenerProperties EMPTY >
<!ATTLIST listenerProperties
      name               (InterwovenOpenDeploy) #FIXED "InterwovenOpenDeploy"
      bindPort           CDATA         #REQUIRED
      >

<!ELEMENT transportProperties EMPTY >
<!ATTLIST transportProperties
      name               (od)          #FIXED "od"
      bufferSize         CDATA         #IMPLIED
      >
<!-- ============================================================
     The version attribute of the teamsiteProperties element
     is used to indicate the TeamSite version that OpenDeploy
     will be communicating for a TeamSite based deployment.
     If the OpenDeploy and TeamSite versions are identical, there
     is no need to include this (the teamsiteProperties) element,
     however, if the TeamSite version number is less than the
     OpenDeploy version, this element and its attribute are
     required to allow OpenDeploy to perform TeamSite based
     deployments.
     ============================================================ -->
<!ELEMENT teamsiteProperties EMPTY >
<!ATTLIST teamsiteProperties
      version            CDATA         #IMPLIED
      >

<!-- ============================================================
     Please read documentation carefully if you decide to alter
     any of the attribute settings used in your odbase.xml or
     odrcvr.xml file from those settings used in the initially
     installed example copy.
     Failure to keep these attributes consistant with one another
     could result in exception error messages being dumped into
     the server log file, and failure of the service to start.
     ============================================================ -->
<!ELEMENT schedulerProperties EMPTY>
<!ATTLIST schedulerProperties
      jdbcDriverClass    CDATA         #IMPLIED
      dbUrl              CDATA         #IMPLIED
      dbUser             CDATA         #IMPLIED
      dbPassword         CDATA         #IMPLIED
```

```
      isClearPassword    (yes|no)      "no"
      >

<!-- ============================================================
     The following restriction rules allow the master server to
     control what is allowed or not allowed to be done.
     ============================================================ -->

<!ELEMENT allowedHosts (node+) >

<!ELEMENT node (allowedDirectories+) >
<!ATTLIST node
      host               CDATA         #REQUIRED
      keyFile            CDATA         #IMPLIED
      >

<!ELEMENT allowedDirectories (path+) >

<!ELEMENT path EMPTY>
<!ATTLIST path
      name               CDATA         #REQUIRED
      >

<!-- ============================================================
     Logfile management
     Note: maxBytes must be specified with a number and an
           indication of the unit.  E.g. "32mb", "22kb".
     Note: the default directory for storing logfiles is
           $IWODHOME/log/.
     ============================================================ -->
<!ELEMENT logRules EMPTY>
<!ATTLIST logRules
      maxBytes          CDATA                          #IMPLIED
      directory         CDATA                          #IMPLIED
      level             (verbose|normal)               "verbose"
      >
```

Descriptions of Elements and Attributes

The following section explains in detail the elements and attributes contained in the deploy server DTD.

deployServerConfiguration

The `deployServerConfiguration` element is the root container for the elements to define the host OpenDeploy server.

DTD Definition

In the deploy server configuration DTD, the `deployServerConfiguration` element is defined as:

```
<!ELEMENT deployServerConfiguration (localNode oldOdHome?
initiatorProperties? listenerProperties transportProperties?
teamsiteProperties schedulerProperties allowedHosts logRules?) >
```

Associated Child Elements

The following child elements are associated with the `deployServerConfiguration` element:

- `localNode` — see page 16.
- `oldOdHome` — see page 17.
- `initiatorProperties` — page 18.
- `listenerProperties` — page 19.
- `transportProperties` — page 19.
- `teamsiteProperties` — page 20.
- `schedulerProperties` — page 21.
- `allowedHosts` — see page 22.
- `logRules` — see page 24.

localNode

The `localNode` element defines the identity of the server host, and encryption methods and values for deployments.

DTD Definition

In the deploy server configuration DTD, the `localNode` element is defined as:

```
<!ELEMENT localNode EMPTY >
<!ATTLIST localNode
   host                    CDATA                    #REQUIRED
   (
      ( sslCertificate     CDATA                    #IMPLIED
        sslPrivateKey      CDATA                    #IMPLIED
        sslCaCertificate   CDATA                    #IMPLIED
        sslCiphers         CDATA                    #IMPLIED
        sslVerifyPeer      (require|request|none)   "none")
   |
   (keyFile                CDATA                    #IMPLIED)
   ) >
```

Associated Attributes

The following attributes are associated with the `localNode` element:

- `host` — specifies the fully qualified DNS host name or IP address of the OpenDeploy server. For example:

  `host="venus.mycompany.com"` *or*

  `host="114.342.23.21"`

- `sslCertificate` — specifies the absolute path to the secure sockets layer (SSL) public key certificate. This attribute is required for using asymmetric key encryption.

- `sslPrivateKey` — specifies the absolute path to the SSL private key certificate. This attribute is required for using asymmetric key encryption.

- `sslCaCertificate` — specifies the absolute path to the certificate authority. This allows OpenDeploy to authenticate the source from which the public and private key pairs for the source and target hosts are derived. This attribute is required for using asymmetric key encryption.

- `sslCiphers` — specifies the SSL ciphers to use. Multiple ciphers must be separated by a colon (":"). For example:

  ```
  sslCiphers="EDH-DSS-DES-CBC3-SHA:EXP-EDH-DSS-DES-CBC-SHA"
  ```

  This attribute is optional for using with asymmetric key encryption.

- `sslVerifyPeer` — indicates which of the following conditions apply in regards to the verification that the certificate authority for each public and private key pairs comes from the same source. This source is the value specified in the `sslCaCertificate` attribute.
  - `none` — no verification is performed. This is the default value.
  - `request` — verification is performed if the certificate/key pair exists on the peer of the host making the authentication request before the deployment can occur.
  - `require` — verification must be performed, and the certificate/key pair must exist on the peer of the host making the request before the deployment can occur.

- `keyFile` — specifies the absolute path to the key file that provides the weak 40-bit symmetric encryption. For example:

  `keyFile="C:\secure\`*`MyKeyFile.txt`*`"` *or*

  `keyFile="/secure/`*`MyKeyFile.txt`*`"`

  This attribute is required for using symmetric key encryption. The `keyFile` attribute is mutually exclusive with the various SSL attributes, as they indicate mutually exclusive encryption methods.

oldOdHome

The `oldOdHome` element defines the location on the host server where OpenDeploy 4.5.2 sending software resides. This element facilitates deploying files to target hosts running the OpenDeploy 4.5.2 receiving software.

DTD Definition

In the deploy server configuration DTD, the `oldOdHome` element is defined as:

```
<!ELEMENT oldOdHome EMPTY >
<!ATTLIST oldOdHome
   path              CDATA          #REQUIRED >
```

Associated Attributes

The following attributes are associated with the `oldOdHome` element:

- `path` — specifies the path to the sending software in the `path` attribute of the `oldOdHome` element in the base server configuration file (by default `odbase.xml`):

  ```
  <oldOdHome path="od452-home" />
  ```

  where *od452-home* is the full path to where the OpenDeploy 4.5.2 sending software resides on the source host. For example:

  ```
  <oldOdHome path="C:\Program Files\Interwoven\OpenDeploy" />
  ```

initiatorProperties

The `initiatorProperties` element defines whether or not the deployment job queuing feature is enabled on the source host.

DTD Definition

In the deploy server configuration DTD, the `initiatorProperties` element is defined as:

```
<!ELEMENT initiatorProperties EMPTY >
<!ATTLIST initiatorProperties
   name              (InterwovenOpenDeploy)     #FIXED
   pendSessions      (yes|no)                   "no" >
```

Associated Attributes

The following attributes are associated with the `initiatorProperties` element:

- `name` — this value is fixed as `InterwovenOpenDeploy`.
- `pendSessions` — indicates whether or not the source host can use deployment job queuing.

listenerProperties

The `listenerProperties` element defines the ports over which source and target hosts communicate with each other and perform OpenDeploy tasks, such as file comparisons and deployments.

DTD Definition

In the deploy server configuration DTD, the `listenerProperties` element is defined as:

```
<!ELEMENT listenerProperties EMPTY >
<!ATTLIST listenerProperties
   name              (InterwovenOpenDeploy)        #FIXED
   bindPort          CDATA                         #REQUIRED >
```

Associated Attributes

The following attributes are associated with the `listenerProperties` element:

- `name` — this value is fixed, but must be supplied, as `InterwovenOpenDeploy`.
- `bindPort` — specifies the port number shared by the source and target hosts.

transportProperties

The `transportProperties` element defines customized buffer settings for sending and receiving files.

DTD Definition

In the deploy server configuration DTD, the `transportProperties` element is defined as:

```
<!ELEMENT transportProperties EMPTY >
<!ATTLIST transportProperties
   name                  (od)            #FIXED
   bufferSize            CDATA           #IMPLIED >
```

Associated Attributes

The following attributes are associated with the `transportProperties` element:

- `name` — this value is fixed, but must be supplied, as `od`.
- `bufferSize` — specifies the buffer size in bytes for sending and receiving deployments. The default value is 8000 bytes (`8000b`).

teamsiteProperties

The `teamsiteProperties` element specifies which version of TeamSite is present in the OpenDeploy environment. This element is only necessary if you want to perform TeamSite-based deployments, and your TeamSite software is of a different release than your OpenDeploy software. If your TeamSite release is the same as your OpenDeploy release, or you do not want to perform TeamSite-based deployments, you can ignore this element.

DTD Definition

In the deploy server configuration DTD, the `teamsiteProperties` element is defined as:

```
<!ELEMENT teamsiteProperties EMPTY >
<!ATTLIST teamsiteProperties
   version               CDATA            #IMPLIED >
```

Associated Attributes

The following attribute is associated with the `teamsiteProperties` element:

- `version` — indicates the TeamSite software release. You only need to include the first two numbers of the TeamSite release. For example:

```
version="5.0"
```

schedulerProperties

The `schedulerProperties` element defines the attributes and values related to a scheduler database.

DTD Definition

In the deploy server configuration DTD, the `schedulerProperties` element is defined as:

```
<!ELEMENT schedulerProperties EMPTY >
<!ATTLIST schedulerProperties
   jdbcDriverClass            CDATA               #IMPLIED
   dbUrl                      CDATA               #IMPLIED
   dbUser                     CDATA               #IMPLIED
   dbPassword                 CDATA               #IMPLIED
   isClearPassword            (yes|no)            "no" >
```

Associated Attributes

The following attributes are associated with the `schedulerProperties` element:

- `jdbcDriverClass` — specifies the JDBC Java class that is used to communicate to the RDBMS. The default value is the Hypersonic SQL database:

  ```
  jdbcDriverClass="org.hsql.jdbcDriver"
  ```

- `dbUrl` — specifies the Web URL to the scheduler database. The default value is:

  ```
  dbUrl="jdbc:HypersonicSQL:od-home/db/schedDB"
  ```

- `dbUser` — specifies the user account name for access to the scheduler database.

- `dbPassword` — specifies the password to the scheduler database. The default value is:

  ```
  dbPassword=""
  ```

- `isClearPassword` — indicates whether or not the value of the `dbPassword` attribute is contained as unencoded plain text in the deployment configuration file. By default, it is assumed that the `dbPassword` value is an encoded string. Default value is `no`. If the `isClearPassword` attribute is value is `yes`, then this password will be in plain text. The Hypersonic SQL database that is installed by default with the base server software does not require a password.

allowedHosts

The `allowedHosts` element specifies from which source hosts a target host can receive deployed files.

DTD Definition

In the deploy server configuration DTD, the `allowedHosts` element is defined as:

```
<!ELEMENT allowedHosts (node+) >
```

Associated Child Elements

The following child element is associated with the `allowedDeployment` element:

- `node` — page 22.

node

The `node` element defines a source host capable of deploying files to this base server or receiver host. It can also specify the `keyFile` attribute for weak 40-bit symmetric encryption.

DTD Definition

In the deploy server configuration DTD, the `node` element is defined as:

```
<!ELEMENT node (allowedDirectories+) >
<!ATTLIST node
   host                 CDATA             #REQUIRED
   keyFile              CDATA             #IMPLIED >
```

Associated Child Elements

The following child element is associated with the `node` element:

- `allowedDirectories` — see page 23.

Associated Attributes

The following attributes are associated with the `node` element:

- `host` — specifies the DNS host name or the IP address of the [remote] source server. For example:

  `host="venus.mycompany.com"` *or*

  `host="114.342.23.21"`

- `keyFile` — specifies the absolute path to the key file for symmetric encryption. For example:

  `keyFile="/local/OpenDeploy/conf/keyfile.txt"`

allowedDirectories

The `allowedDirectories` element defines which directory locations on the local host are allowed to be used for receiving files as part of a deployment.

DTD Definition

In the deploy server configuration DTD, the `allowedDirectories` element is defined as:

```
<!ELEMENT allowedDirectories (path+) >
```

Associated Child Elements

The following child element is associated with the `allowedDeployment` element:

- `path` — see page 24.

path

The `path` element defines the absolute path to an area on the local host under which it is permissible to deploy content.

DTD Definition

In the deploy server configuration DTD, the `path` element is defined as:

```
<!ELEMENT path EMPTY>
<!ATTLIST path
   name           CDATA             #REQUIRED >
```

Associated Attributes

The following attribute is associated with the `path` element:

- `name` — specifies the actual name of the `path` element. For Windows, the value:

  ```
  name="x:\"
  ```

  by itself, where `x` is a drive letter, is not an accepted path.

logRules

The `logRules` element defines values for log management features.

DTD Definition

In the deploy server configuration DTD, the `logRules` element is defined as:

```
<!ELEMENT logRules EMPTY >
<!ATTLIST logRules
   maxBytes            CDATA                  #IMPLIED
   directory           CDATA                  #IMPLIED
   level               (verbose|normal)       "verbose" >
```

Associated Attributes

The following attributes are associated with the `logRules` element:

- `maxBytes` — specifies the maximum size in bytes a log file is allowed to grow before the file is closed and OpenDeploy begins writing to a new file. This value is known as the *rollover threshold*. The default `maxBytes` value is 32 megabytes. You can specify different byte measurements in the value, including megabytes (`mb`), kilobytes (`kb`), and bytes (`b`). For example:

  `maxBytes="10mb"` *or*

  `maxBytes="10000kb"` *or*

  `maxBytes="10000000b"`

  indicates that the log file size can grow to 10 megabytes before OpenDeploy will close that log file and start a new one.

  Ensure that you include the proper measurement indicator when setting the threshold size. If no recognizable size measurement is indicated, OpenDeploy uses its default value instead. For example, if the following value was specified:

  `maxBytes="10"`

  OpenDeploy would ignore that stated value and use the default value (`32mb`) instead.

  If the unit of measure is present but unrecognized by OpenDeploy, the default value is used. For example, if the following value was specified:

  `maxBytes="1000x"`

  OpenDeploy would ignore this value and use the default value (`32mb`).

  OpenDeploy will not honor a `maxBytes` value of less than 100 kilobytes (`100kb`). For example, if the following value was specified:

  `maxBytes="50kb"`

  OpenDeploy would ignore this value and use the default value (`32mb`) instead.

- `directory` — specifies the absolute path directory location for log files. The default location is:

  od-home`/log`

- `level` – indicates the level and type of logging OpenDeploy will perform.
  - `verbose` — logs high level of detail on deployment events as they occur. This logging level is best suited for troubleshooting deployment problems or evaluating deployment performance. Verbose logging can create very large log files. This is the default logging level.
  - `normal` — logs standard status and error messages. In most cases, this level of logging provides a sufficient amount of detail to meet your needs.

Chapter 2

Nodes Configuration DTD

The nodes configuration DTD defines the elements and attributes for the nodes configuration file, which specifies the target hosts available to the source host.

Nodes Configuration DTD

This section contains the XML code contained within the nodes configuration DTD.

```
<!-- ============================================================ -->
<!--                Nodes Configuration DTD                       -->
<!-- ============================================================ -->

<!-- ============================================================
     A nodeSet defines all OpenDeploy servers (senders or
     receivers) known to this OpenDeploy server.
     ============================================================ -->
<!ELEMENT nodeSet (node+) >

<!-- ============================================================
     A node represents an individual OpenDeploy server:
     ____________________________________________________________
     - name: a symbolic name by which to refer to the node.
     - host: the actual hostname or IP address of the node.
     - port: the port number to use for communicating with the
             node.
     ============================================================ -->
<!ELEMENT node EMPTY >
<!ATTLIST node
      name           ID            #REQUIRED
      host           CDATA         #REQUIRED
      port           CDATA         #REQUIRED
      >
```

Descriptions of Elements and Attributes

This section describes each element and its associated attributes found within the nodes configuration DTD.

nodeSet

The `nodeSet` element is the root container for elements representing all target nodes to which the source host can deploy files.

DTD Definition

In the nodes configuration DTD, the `nodeSet` element is defined as:

```
<!ELEMENT nodeSet (node)+ >
```

Associated Child Elements

The following child element is associated with the `nodeSet` element:

- `node` — see below.

node

The `node` element defines a target host capable of receiving files from the OpenDeploy source host.

DTD Definition

In the nodes configuration DTD, the `node` element is defined as:

```
<!ELEMENT node EMPTY >
<!ATTLIST node
   name           ID              #REQUIRED
   host           CDATA           #REQUIRED
   port           CDATA           #REQUIRED >
```

Associated Attributes

The following attributes are associated with the `node` element:

- `name` — the logical name of the host as it appears in OpenDeploy configuration files. For example:

  ```
  name="venus"
  ```

- `host` — the fully qualified DNS host name or the IP address of the server. For example:

  ```
  host="venus.mycompany.com"
  ```
  or

  ```
  host="114.342.23.21"
  ```

- `port` — the port number used by the OpenDeploy host. The port number should match the value for the `bindPort` attribute of the `listenerProperties` element found in the base server or receiver configuration file of the target host. For example:

  ```
  port="20014"
  ```

Chapter 3

Deployment Configuration DTD

The deployment configuration DTD specifies the XML rules for deployment configuration files. Each named deployment configuration has a corresponding deployment configuration file that defines the type of deployment taking place, the target hosts, and what OpenDeploy features are being employed.

Deployment Configuration DTD

This section contains the XML code contained within the deployment configuration DTD.

```
<!-- ============================================================ -->
<!--                 Deployment Configuration DTD                 -->
<!-- ============================================================ -->

<!ELEMENT deploymentConfiguration  ( logRules?,
                                     localNode,
                                     replicationFarmSet,
                                     definition+,
                                     deployment
                                   ) >

<!-- ============================================================
     A deployment configuration has dependencies on a set of
     machines (nodes) defined in IWODHOME/etc/odnodes.xml
     ============================================================
     NOTE: The following specification is syntactically correct
           as far as XML is concerned, but not expressive enough
           for the purposes of OpenDeploy.
           Thus, please be aware that while all the ssl*
           attributes and keyFile attribute are optional - if you
           decide to use the ssl* attributes you MUST use ALL of
           them and CANNOT use the keyFile attribute.
           Conversely, if you decide to use the keyFile attribute,
           you CANNOT use ANY of the ssl* attributes.
     ============================================================ -->
```

```
<!ELEMENT localNode EMPTY >
<!ATTLIST localNode
      host                CDATA      #REQUIRED
      sslCertificate      CDATA      #IMPLIED
      sslPrivateKey       CDATA      #IMPLIED
      sslCaCertificate    CDATA      #IMPLIED
      sslCiphers          CDATA      #IMPLIED
      sslVerifyPeer       (require|request|none)        "none"
      keyFile             CDATA      #IMPLIED
      >

<!-- ============================================================
     A deployment definition contains a pairing of "source" and
     "target"
     ============================================================ -->

<!ELEMENT definition ( (source, target)|(reverseSource, reverseTarget) ) >
<!ATTLIST definition
      name              ID          #REQUIRED
      >

<!-- ============================================================
     A deployment itself specifies multiple source-target tasks.
     ============================================================ -->
<!ELEMENT deployment ( execDeploymentTask+ ) >
<!ATTLIST deployment
      transactional     (yes|no)           "no"
      >

<!-- ============================================================
     The "downRev" attribute is required when you are deploying
     from an OpenDeploy 5.5.x to an OpenDeploy 4.5.x system.  The
     value of the attribute should be the version number of the
     receiving OpenDeploy server.
     ============================================================ -->
<!ELEMENT execDeploymentTask (deployNRun?) >
<!ATTLIST execDeploymentTask
      useDefinition     IDREF       #REQUIRED
      downRev           CDATA       #IMPLIED
      >

<!-- ============================================================
     The "source" and "target" of a deployment are
     the data-sources and data-receptors, respectively.
     Note that there may be multiple data-sources defined.
```

```
     ============================================================ -->

<!ELEMENT source ( (sourceFilesystem | sourceTeamsite)+ ) >

<!ELEMENT target  (targetFilesystem, filters?, comparisonRules?,
                   transferRules?, permissionRules?)
                  >
<!ATTLIST target
      useReplicationFarm IDREF       #REQUIRED
      >

<!-- ============================================================
     To define a reverse deployment, use the following elements.
     The "reverseTarget" and "reverseSource" of a deployment are
     the data-receptors and data-sources, respectively.
     ============================================================ -->

<!ELEMENT reverseSource ( (sourceFilesystem | sourceTeamsite)+ ) >
<!ATTLIST reverseSource
      useReplicationFarm IDREF       #REQUIRED
      >

<!ELEMENT reverseTarget  (targetFilesystem, filters?, comparisonRules?,
                          transferRules?, permissionRules?)
                         >

<!-- ============================================================
     These adapters support the core OpenDeploy functionality.
     The sourceFilesystem adapter specifies the source location
         for the content on a file system.
     The targetFilesystem adapter specifies where it will
         be transported to.
     The sourceTeamsite adapter specifies the teamsite area(s)
         that will be differenced and then deployed over.
     ============================================================ -->

<!ELEMENT sourceFilesystem ( pathSpecification+ ) >
<!ATTLIST sourceFilesystem
      name              CDATA        #IMPLIED
      area              CDATA        #REQUIRED
      filelist          CDATA        #IMPLIED
      >

<!ELEMENT sourceTeamsite ( pathSpecification+ ) >
<!ATTLIST sourceTeamsite
```

```
      name              CDATA        #IMPLIED
      area              CDATA        #REQUIRED
      previousArea      CDATA        #REQUIRED
      >

<!ELEMENT targetFilesystem EMPTY >
<!ATTLIST targetFilesystem
      area              CDATA        #REQUIRED
      >

<!-- ============================================================
     The following sections specify how to configure options for
     particular adapter elements
     ============================================================ -->

<!ELEMENT pathSpecification ( path,
                              filters?,
                              sourceTransferRules?,
                              targetRules?
                            ) >

<!-- ============================================================
     Note: targetRules settings within pathSpecification will
           override the settings in target.
     Note: targetRules settings within nodeRef will override
           the settings in both pathSpecification *and* target.
     ============================================================ -->
<!ELEMENT targetRules (filters?, comparisonRules?, transferRules?,
                       permissionRules?) >
<!ATTLIST targetRules
      area              CDATA        #IMPLIED
      >

<!ELEMENT path EMPTY>
<!ATTLIST path
      name              CDATA        #REQUIRED
      >

<!ELEMENT filters ( (excludePath | excludePattern)+ ) >

<!ELEMENT excludePath EMPTY >
<!ATTLIST excludePath
      subPath           CDATA        #REQUIRED
      >
```

```
<!ELEMENT excludePattern EMPTY >
<!ATTLIST excludePattern
      regex             CDATA        #REQUIRED
      >

<!-- ============================================================
     Note: dateDifferent and revert cannot both be enabled for
           the same deployment.
     ============================================================ -->
<!ELEMENT comparisonRules EMPTY >
<!ATTLIST comparisonRules
      dateDifferent     (yes|no)     "no"
      revert            (yes|no)     "no"
      ignoreAcls        (yes|no)     "no"
      ignoreModes       (yes|no)     "no"
      ignoreUser        (yes|no)     "no"
      ignoreGroup       (yes|no)     "no"
      >

<!ELEMENT transferRules EMPTY >
<!ATTLIST transferRules
      doDeletes                 (yes|no)    "no"
      dontDo                    (yes|no)    "no"
      preserveAcls              (yes|no)    "no"
      followLinks               (yes|no)    "no"
      svrTryCount               CDATA       #IMPLIED
      svrTryInterval            CDATA       #IMPLIED
      svrTryDisableOverwrite    (yes|no)    "no"
      rmReadOnly                (yes|no)    "no"
      >

<!ELEMENT sourceTransferRules EMPTY >
<!ATTLIST sourceTransferRules
      followLinks       (yes|no)     "no"
      >

<!ELEMENT permissionRules (userTranslation|groupTranslation)* >
<!ATTLIST permissionRules
      amask             CDATA        #IMPLIED
      omask             CDATA        #IMPLIED
      directory         CDATA        #IMPLIED
      file              CDATA        #IMPLIED
      group             CDATA        #IMPLIED
      user              CDATA        #IMPLIED
      changeAccess      CDATA        #IMPLIED
```

```
      setAccess         CDATA        #IMPLIED
      >

<!ELEMENT userTranslation EMPTY >
<!ATTLIST userTranslation
      from              CDATA        #REQUIRED
      to                CDATA        #REQUIRED
      >

<!ELEMENT groupTranslation EMPTY >
<!ATTLIST groupTranslation
      from              CDATA        #REQUIRED
      to                CDATA        #REQUIRED
      >

<!-- ============================================================
     The deployNRun feature specifies when and how a script is
     invoked
     ============================================================ -->

<!ELEMENT deployNRun ( (dnrFile | dnrDir | dnrDeployment)+ ) >

<!-- ============================================================
     Note: dnrFile does not actually support location="source".
           At this time, specifying such will cause the
           specification to be silently ignored as a whole.
     ============================================================ -->
<!ELEMENT dnrFile ( script ) >
<!ATTLIST dnrFile
      location          (target)          #REQUIRED
      when              (before | after)  #REQUIRED
      state             (success | failure | always)  "always"
      mask              CDATA             #IMPLIED
      >

<!-- ============================================================
     Note: dnrDir does not actually support location="source".
           At this time, specifying such will cause the
           specification to be silently ignored as a whole.
     ============================================================ -->
<!ELEMENT dnrDir ( script ) >
<!ATTLIST dnrDir
      location          (target)           #REQUIRED
      when              (before | after)   #REQUIRED
      state             (success | failure | always)  "always"
```

```
      mask              CDATA              #IMPLIED
      >

<!ELEMENT dnrDeployment ( script ) >
<!ATTLIST dnrDeployment
      location          (source | target)  #REQUIRED
      when              (before | after)   #REQUIRED
      state             (success | failure | always)  "always"
      >

<!ELEMENT script EMPTY >
<!ATTLIST script
      cmd               CDATA        #REQUIRED
      as                CDATA        #IMPLIED
      where             CDATA        #IMPLIED
      async             (yes|no)    "no"
      >

<!-- ============================================================
     Logfile management
     Note: maxBytes must be specified with a number and an
           indication of the unit.  E.g. "32mb", "22kb".
     ============================================================ -->

<!ELEMENT logRules EMPTY >
<!ATTLIST logRules
      maxBytes          CDATA                          #IMPLIED
      level             (verbose|normal)               "verbose"
      >

<!-- ============================================================
     Nodes are machines in a network defined in a well-known file.

     Nodes can be annotated with a reference to a subsequent
     deployment and configuration and when that deployment is
     to be triggered after successful completion of the current
     node.
     ============================================================ -->

<!ELEMENT nodeRef (nextDeployment*, targetRules?) >
<!ATTLIST nodeRef
      useNode           IDREF        #REQUIRED
      >

<!ELEMENT nextDeployment EMPTY>
```

```
<!ATTLIST nextDeployment
      deployment        CDATA        #REQUIRED
      invokeOnSuccess   (yes|no)     "no"
      >

<!ELEMENT replicationFarmSet (replicationFarm+) >

<!ELEMENT replicationFarm (nodeRef+) >
<!ATTLIST replicationFarm
      name              ID           #REQUIRED
      >
```

Descriptions of Elements and Attributes

The following section explains the elements and attributes contained in the deployment configuration DTD.

Elements at a Glance

The elements in this appendix follow in the order they appear in the deployment configuration DTD. However, the following table lists them in alphabetical order with their corresponding page number.

deploymentConfiguration

The `deploymentConfiguration` element defines the overall deployment, acting as the root container for the deployment configuration's various elements and attributes.

DTD Definition

In the deployment configuration DTD, the `deploymentConfiguration` element is defined as:

```
<!ELEMENT deploymentConfiguration (logRules? localNode
replicationFarmSet definition+ deployment) >
```

Associated Child Elements

The following child elements are associated with the `deploymentConfiguration` element:

- `logRules` — see page 69.
- `localNode` — see page 41.
- `replicationFarmSet` — see page 72.
- `definition` — see page 43.
- `deployment` — see page 44.

localNode

The `localNode` element defines host name and encryption information regarding the source host of the deployment.

DTD Definition

In the deployment configuration DTD, the `localNode` element is defined as:

```
<!ELEMENT localNode EMPTY >
<!ATTLIST localNode
   host                CDATA                    #REQUIRED
   ( (sslCertificate   CDATA                    #IMPLIED
      sslPrivateKey    CDATA                    #IMPLIED
      sslCaCertificate CDATA                    #IMPLIED
      sslCiphers       CDATA                    #IMPLIED
      sslVerifyPeer    (require|request|none)   "none") |
   (keyFile            CDATA                    #IMPLIED) ) >
```

Associated Attributes

The following attributes are associated with the `localNode` element:

- `host` — specifies the fully qualified DNS host name or IP address of the source host. For example:

  `host="venus.mycompany.com"` *or*

  `host="114.342.23.21"`

- `sslCertificate` — specifies the absolute path to the secure sockets layer (SSL) public key certificate. This attribute is required for using asymmetric key encryption.

- `sslPrivateKey` — specifies the absolute path to the SSL private key certificate. This attribute is required for using asymmetric key encryption.

- `sslCaCertificate` — specifies the absolute path to the certificate authority. This allows OpenDeploy to authenticate the source from which the public and private key pairs for the source and target hosts are derived. This attribute is required for using asymmetric key encryption.

- `sslCiphers` — specifies the SSL ciphers to use. Multiple ciphers must be separated by colons (":"). For example:

  ```
  sslCiphers="EDH-DSS-DES-CBC3-SHA:EXP-EDH-DSS-DES-CBC-SHA"
  ```

  This attribute is optional for using with asymmetric key encryption.

- `sslVerifyPeer` — indicates which of the following conditions apply in regards to the verification that the certificate authority for each public and private key pairs comes from the same source. This source is the value specified in the `sslCaCertificate` attribute.
  - `none` — no verification is performed. This is the default value.
  - `request` — verification is performed if the certificate/key pair exists on the peer of the host making the authentication request before the deployment can occur.
  - `require` — verification must be performed, and the certificate/key pair must exist on the peer of the host making the request before the deployment can occur.

- `keyFile` — specifies the absolute path to the key file that provides the weak 40-bit symmetric encryption. For example:

  `keyFile="C:\secure\`*`MyKeyFile.txt`*`"` *or*

  `keyFile="/secure/`*`MyKeyFile.txt`*`"`

  This attribute is required for using symmetric key encryption. The `keyFile` attribute is mutually exclusive with the various SSL attribute, as they indicate mutually exclusive encryption methods.

definition

The `definition` element specifies the matching between one or more source file locations (either a file system location or TeamSite area) and a single target file system location in a deployment. Within the definition are the various deployment criteria rules and features that determine which files can be deployed.

DTD Definition

In the deployment configuration DTD, the `definition` element is defined as:

```
<!ELEMENT definition ( (source, target) | (reverseSource,
reverseTarget) ) >
<!ATTLIST definition
   name           ID          #REQUIRED >
```

Associated Child Elements

The following child elements are associated with the `definition` element:

- `source` — see page 45.
- `target` — see page 46.
- `reverseSource` — see page 47.
- `reverseTarget` — see page 48.

Associated Attributes

The following attribute is associated with the `definition` element:

- `name` — denotes the unique name of the `definition` element.

deployment

The `deployment` element specifies the various source-target tasks involved in a deployment.

DTD Definition

In the deployment configuration DTD, the `deployment` element is defined as:

```
<!ELEMENT deployment (execDeploymentTask+) >
<!ATTLIST deployment
   transactional        (yes|no)          "no" >
```

Associated Child Elements

The following child element is associated with the `deployment` element:

- `execDeploymentTask` — see page 44.

Associated Attributes

The following attribute is associated with the `deployment` element:

- `transactional` — indicates whether or not the deployment configuration is transactional. Default value is `no`.

execDeploymentTask

The `execDeploymentTask` element specifies which deployment definition to run.

DTD Definition

In the deployment configuration DTD, the `execDeploymentTask` element is defined as:

```
<!ELEMENT execDeploymentTask (deployNRun?) >
<!ATTLIST execDeploymentTask
   useDefinition        IDREF       #REQUIRED
   downRev              CDATA       #IMPLIED >
```

Associated Child Elements

The following child element is associated with the `execDeploymentTask` element:

- `deployNRun` — see page 64.

Associated Attributes

The following attributes are associated with the `execDeploymentTask` element:

- `useDefinition` — points to a specific named `definition` element attribute previously defined.
- `downRev` — specifies that the receiving nodes are running an earlier version of OpenDeploy, and which release it is. For example:

```
downRev="4.5.2"
```

source

The `source` element identifies attributes regarding the originator of the deployment.

DTD Definition

In the deployment configuration DTD, the `source` element is defined as:

```
<!ELEMENT source ( (sourceFilesystem | sourceTeamsite)+ ) >
```

Associated Child Elements

The following child elements are associated with the `source` element:

- `sourceFilesystem` — see page 48.
- `sourceTeamsite` — see page 49.

target

The `target` element identifies attributes regarding the recipient of the deployment. Note that `targetRules` settings specified within the `pathSpecification` element will override settings within the `target` element.

DTD Definition

In the deployment configuration DTD, the `target` element is defined as:

```
<!ELEMENT target ( targetFilesystem filters? comparisonRules?
transferRules? permissionRules?) >
<!ATTLIST target
   useReplicationFarm   IDREF       #IMPLIED >
```

Associated Child Elements

The following child elements are associated with the `target` element:

- `targetFilesystem` — see page 50.
- `filters` — see page 54.
- `comparisonRules` — see page 56.
- `transferRules` — see page 58.
- `permissionRules` — page 60.

Associated Attribute

The following attribute is associated with the `target` element:

- `useReplicationFarm` — points to a specific named `replicationFarm` element attribute previously defined. For example, if you previously defined a named `replicationFarm` element as `fan-out`, you can reference that element here as:

  ```
  useReplicationFarm="fan-out"
  ```

reverseSource

The `reverseSource` element identifies the attributes regarding the originator of a reverse deployment.

DTD Definition

In the deployment configuration DTD, the `reverseSource` element is defined as:

```
<!ELEMENT reverseSource ( (sourceFilesystem | sourceTeamsite)+ ) >
<!ATTLIST reverseSource
   useReplicationFarnm        IDREF       #IMPLIED >
```

Associated Child Elements

The following child elements are associated with the `reverseSource` element:

- `sourceFilesystem` — see page 48.
- `sourceTeamsite` — see page 49.

Associated Attributes

The following attribute is associated with the `reverseSource` element:

- `useReplicationFarm` — points to a specific named `replicationFarm` element attribute previously defined. For example, if you previously defined a named `replicationFarm` element as `fan-out`, you can reference that element here as:

  ```
  useReplicationFarm="fan-out"
  ```

reverseTarget

The `reverseTarget` element identifies the attributes regarding the recipient of a reverse deployment.

DTD Definition

In the deployment configuration DTD, the `reverseTarget` element is defined as:

```
<!ELEMENT reverseTarget (targetFilesystem filters? comparisonRules?
transferRules? permissionRules?) >
```

Associated Child Elements

The following child elements are associated with the `reverseTarget` element:

- `targetFilesystem` — see page 50.
- `filters` — see page 54.
- `comparisonRules` — see page 56.
- `transferRules` — see page 58.
- `permissionRules` — page 60.

sourceFilesystem

The `sourceFilesystem` element defines the attributes for a file system-based repository containing the source host files.

DTD Definition

In the deployment configuration DTD, the `sourceFilesystem` element is defined as:

```
<!ELEMENT sourceFilesystem (pathSpecification+) >
<!ATTLIST sourceFilesystem
   name                 CDATA            #IMPLIED
   area                 CDATA            #REQUIRED
   filelist             CDATA            #IMPLIED >
```

Associated Child Elements

The following child element is associated with the `sourceFilesystem` element:

- `pathSpecification` — see page 51.

Associated Attributes

The following attributes are associated with the `sourceFilesystem` element:

- `name` — denotes the unique name of the `sourceFilesystem` element.
- `area` — specifies the absolute path for a file system containing the source host files. For example:

  ```
  area="/website/files"
  ```
  or
  ```
  area="C:\website\files"
  ```

- `filelist` — specifies the absolute path to the file being accessed in a file list deployment. The presence of this attribute and value indicate the deployment is a file list type. The absence of this attribute indicates the deployment is a directory comparison type. For example:

  ```
  filelist="C:\OpenDeploy\files\filelist.txt"
  ```

sourceTeamsite

The `sourceTeamsite` element defines the attributes for a TeamSite area-based repository containing the originating host files.

DTD Definition

In the deployment configuration DTD, the `sourceTeamsite` element is defined as:

```
<!ELEMENT sourceTeamsite (pathSpecification+) >
<!ATTLIST sourceTeamsite
   name                 CDATA             #IMPLIED
   area                 CDATA             #REQUIRED
   previousArea         CDATA             #REQUIRED >
```

Associated Child Elements

The following child element is associated with the `sourceTeamsite` element:

- `pathSpecification` — see page 51.

Associated Attributes

The following attributes are associated with the `sourceTeamsite` element:

- `name` — denotes the unique name of the `sourceTeamsite` element.
- `area` — specifies the absolute path for a TeamSite area containing the source host files. For example:

  ```
  area="//IWSERVER/default/main/dev/EDITION"
  ```

- `previousArea` — specifies the absolute path for a TeamSite area containing the target host files in their current state. For example:

  ```
  previousArea="//IWSERVER/default/main/dev/EDITION/IW_PREV"
  ```

targetFilesystem

The `targetFilesystem` element defines the attributes for a file system-based repository containing the recipient host files.

DTD Definition

In the deployment configuration DTD, the `targetFilesystem` element is defined as:

```
<!ELEMENT targetFilesystem Empty >
<!ATTLIST targetFilesystem
   area              CDATA               #REQUIRED >
```

Associated Attributes

The following attributes are associated with the `targetFilesystem` element:

- `area` — specifies the absolute path for a file system containing the target host files. For example:

  ```
  area="/website/files" or

  area="C:\website\files"
  ```

pathSpecification

The `pathSpecification` element is a container of source elements that define deployment rules such as filtering and specifying locations within the source host area. Note that `targetRules` settings specified within the `pathSpecification` element will override settings within the `target` element. Additionally, `targetRules` settings within the `nodeRef` element will override the settings in both the `pathSpecification` and `target` elements.

DTD Definition

In the deployment configuration DTD, the `pathSpecification` element is defined as:

```
<!ELEMENT pathSpecification (path filters? sourceTransferRules?
targetRules?) >
```

Associated Child Elements

The following child elements are associated with the `pathSpecification` element:

- `path` — see page 53.
- `filters` — see page 54.
- `sourceTransferRules` — see page 59.
- `targetRules` — see page 52.

targetRules

The `targetRules` element defines target node modifications to, or exemptions from, configuration standards in multi-target deployments. Note that `targetRules` settings specified within the `pathSpecification` element will override settings within the `target` element. Additionally, `targetRules` settings within the `nodeRef` element will override the settings in both the `pathSpecification` and `target` elements.

DTD Definition

In the deployment configuration DTD, the `targetRules` element is defined as:

```
<!ELEMENT targetRules (filters? transferRules? comparisonRules?
permissionRules?) >
<!ATTLIST targetRules
   area                 CDATA             #IMPLIED >
```

Associated Child Elements

The following child elements are associated with the `targetRules` element:

- `filters` — see page 54.
- `transferRules` — see page 58.
- `comparisonRules` — see page 56.
- `permissionRules` — see page 60.

Associated Attributes

The following attribute is associated with the `targetRules` element:

- `area` — specifies an alternate file system location on the target node where the deployed files will reside.

path

The `path` element defines a location within a specified file system location or TeamSite area as specified in the `area` attribute.

DTD Definition

In the deployment configuration DTD, the `path` element is defined as:

```
<!ELEMENT path EMPTY >
<!ATTLIST path
   name           CDATA             #REQUIRED >
```

Associated Attributes

The following attribute is associated with the `path` element:

- `name` — denotes the path to the subdirectory within the associated `area` attribute (file system location or TeamSite area). For example, if you specified the following value:

  ```
  name="western"
  ```

  where the associated area attribute value is:

  ```
  area="C:\website\files"
  ```

  then the file location path would be:

  ```
  C:\website\files\western
  ```

  If no subdirectory is desired, enter the value:

  ```
  name="."
  ```

filters

The `filters` element defines whether certain paths or data patterns should be included with or excluded from the deployment. Filters can be applied either to files residing at the source-side of a deployment, or at the target-side.

DTD Definition

In the deployment configuration DTD, the `filters` element is defined as:

```
<!ELEMENT filters ( (excludePath | excludePattern)+ ) >
```

Associated Child Elements

The following child elements are associated with the `filters` element:

- `excludePath` — see page 54.
- `excludePattern` — see page 55.

excludePath

The `excludePath` element defines a path that should be excluded from either the source or target hosts during a deployment based upon where the containing filters element is located.

DTD Definition

In the deployment configuration DTD, the `excludePath` element is defined as:

```
<!ELEMENT excludePath EMPTY >
<!ATTLIST excludePath
   subPath              CDATA             #REQUIRED >
```

Associated Attributes

The following attribute is associated with the `excludePath` element:

- `subPath` — specifies the path to a location whose files are excluded from the deployment based upon where the containing filters element is located.

  On the source side, filters exclude files from being deployed and from appearing to exist during the comparison stage (if any) such that if the `doDeletes` attribute is enabled, the files will be removed from the target node.

  On the target side, filters exclude files from appearing to exist during the comparison stage (if any) such that if the files exist on the source side, they will always be deployed.

  The `subPath` value is relative to the `area` attribute with which the filter is associated. In the following example:

  ```
  subPath="monthly"
  ```

  the directory `monthly` would be excluded from the deployment.

excludePattern

The `excludePattern` element defines a regular expression pattern that should be excluded from either the source or target hosts during a deployment based upon where the containing filters element is located.

DTD Definition

In the deployment configuration DTD, the `excludePattern` element is defined as:

```
<!ELEMENT excludePattern EMPTY >
<!ATTLIST excludePattern
   regex             CDATA                #REQUIRED >
```

Associated Attributes

The following attribute is associated with the `excludePattern` element:

- `regex` — specifies a regular expression specifying the pattern to be excluded in the deployment. The items compared with the patterns are paths relative to the target directory. In the following example:

  ```
  regex="internal"
  ```

  any path containing the term `internal` would be excluded from the deployment.

comparisonRules

The `comparisonRules` element defines the rules that OpenDeploy uses when it compares files to determine the eligibility of files for deployment. The type and platform of the file system determines the criteria available. Note that the `dateDifferent` attribute cannot be enabled if the `revert` attribute is also enabled.

DTD Definition

In the deployment configuration DTD, the `comparisonRules` element is defined as:

```
<!ELEMENT comparisonRules EMPTY >
<!ATTLIST comparisonRules
   (
      (dateDifferent       (yes|no)            "no") |
      (revert              (yes|no)            "no")
   )
   ignoreAcls              (yes|no)            "no"
   ignoreModes             (yes|no)            "no"
   ignoreUser              (yes|no)            "no"
   ignoreGroup             (yes|no)            "no" >
```

Associated Attributes

The following attributes are associated with the `comparisonRules` element:

- `dateDifferent` — indicates whether or not a file is deployable if there is any difference in file date (older or newer) between the source and target versions. This differs from the OpenDeploy default date-based comparison setting, where a file is deployed only if the source file is newer than the target file. A value of `yes` indicates that the file should be deployed. Default value is `no`. The `dateDifferent` attribute cannot be enabled if the `revert` attribute is also enabled.
- `revert` — indicates whether or not a file is deployable if the source version is older than the target version. A value of `yes` indicates that the file should be deployed. Default value is `no`. The `revert` attribute cannot be enabled if the `dateDifferent` attribute is also enabled.
- `ignoreAcls` (Windows only) — indicates whether or not to ignore differences in the Windows access control lists (ACLs) during the file comparison. Default value is `no`.
- `ignoreModes` (UNIX only) — indicates whether or not to ignore differences in the UNIX-based permission bit mask during the file comparison. Default value is `no`.
- `ignoreUser` (UNIX only) — indicates whether or not to ignore differences in the UNIX-based file user ownership during the file comparison. Default value is `no`.
- `ignoreGroup` (UNIX only) — indicates whether or not to ignore differences in the UNIX-based file group ownership during the file comparison. Default value is `no`.

transferRules

The `transferRules` element defines the rules for moving files from the source host to the target host during the deployment.

DTD Definition

In the deployment configuration DTD, the `transferRules` element is defined as:

```
<!ELEMENT transferRules EMPTY >
<!ATTLIST transferRules
   doDeletes                    (yes|no)          "no"
   dontDo                       (yes|no)          "no"
   preserveAcls                 (yes|no)          "no"
   followLinks                  (yes|no)          "no"
   svrTryCount                  CDATA             #IMPLIED
   svrTryInterval               CDATA             #IMPLIED
   svrTryDisableOverwrite       (yes|no)          "no"
   rmReadOnly                   (yes|no)          "no" >
```

Associated Attributes

The following attributes are associated with the `transferRules` element:

- `doDeletes` — indicates whether or not files and directories not present in the source host area will be deleted on the target host. Default value is `no`.
- `dontDo` — indicates whether or not to proceed with the deployment following the comparison. Deployment will not occur if this attribute is enabled. This is a good tool to use to check and compare files without actually performing a deployment. Default value is `no`.
- `preserveAcls` (Windows only) — indicates whether or not to preserve the Windows access control lists (ACLs) when the files are moved. By default, OpenDeploy applies ACLs based on the ACLs already existing on the containing folders on the target host receiving the deployed files. Default value is `no`.
- `followLinks` (UNIX only) — indicates whether or not symbolic links on the target hosts will be followed when the files are moved. Default value is `no`.

- `svrTryCount` (Windows only) — specifies the number of times OpenDeploy will attempt to deploy the file to the target host. This feature works in conjunction with Microsoft IIS, and is designed to accommodate times of heavy production server traffic.
- `svrTryInterval` (Windows only) — specifies the amount of time in seconds OpenDeploy waits between deployment attempts. This feature works in conjunction with Microsoft IIS, and is designed to accommodate times of heavy production server traffic.
- `svrTryDisableOverwrite` (Windows only) — indicates whether or not to disable the ability of OpenDeploy to deploy files to a server even if the `svrTryCount` and `svrTryInterval elements` are specified. This feature works in conjunction with Microsoft IIS, and is designed to accommodate times of heavy production server traffic. Default value is `no`.
- `rmReadOnly` (Windows only) — indicates whether or not you want a deployed file to be able to overwrite its read-only target equivalent. If this feature is enabled with a value of `yes`, OpenDeploy will remove the read-only attribute from the target file, allowing the deployment to occur. A value of no will prevent the overwriting. Default value is `no`.

sourceTransferRules

The `sourceTransferRules` element defines the rules applicable to the source host for transferring files during a deployment.

DTD Definition

In the deployment configuration DTD, the `sourceTransferRules` element is defined as:

```
<!ELEMENT sourceTransferRules EMPTY >
<!ATTLIST sourceTransferRules
   followLinks             (yes|no)          "no" >
```

Associated Attributes

The following attribute is associated with the `sourceTransferRules` element:

- `followLinks` — indicates whether or not symbolic links on the source and target hosts will be retained and followed once the files are moved. Default value is `no`.

permissionRules

The `permissionRules` element defines the rules applicable to the permissions of deployed files and directories.

DTD Definition

In the deployment configuration DTD, the `permissionRules` element is defined as:

```
<!ELEMENT permissionRules (userTranslation* groupTranslation*) >
<!ATTLIST permissionRules
   amask                    CDATA                #IMPLIED
   omask                    CDATA                #IMPLIED
   directory                CDATA                #IMPLIED
   file                     CDATA                #IMPLIED
   group                    CDATA                #IMPLIED
   user                     CDATA                #IMPLIED
   changeAccess             CDATA                #IMPLIED
   setAccess                CDATA                #IMPLIED >
```

Associated Child Elements

The following child elements are associated with the `permissionRules` element:

- `userTranslation` — see page 62.
- `groupTranslation` — see page 63.

Associated Attributes

The following attributes are associated with the `permissionRules` element:

- `amask` (UNIX only) — specifies the bit mask (in octal) to be ANDed with the permission bits of all files and directories. The `amask` octal value combines with the existing permission bit value of the affected file. If a file has the existing permission value of `664` (-rw-rw-r--) and the `amask` attribute as the following value:

  ```
  amask="770"
  ```

  then the resulting permission for that file (664 AND 770) following the deployment would be `660` (-rw-rw----).

- `omask` (UNIX only) — specifies the bit mask (in octal) to be ORed with the permission bits of all files and directories. The `omask` octal value combines with the existing permission bit value of the affected file. If a file has the existing permission value of `666` (-rw-rw-rw-) and the `omask` attribute as the following value:

  ```
  omask="022"
  ```

  then the resulting permission for that file (666 OR 022) following the deployment would be `644` (-rw-r--r--).

- `directory` (UNIX only) — specifies the permissions (in octal) given to all deployed directories. For example, if you wanted deployed directories to have the permission "drwxrwx---", then the resulting value would be:

  ```
  directory="770"
  ```

- `file` (UNIX only) — specifies the permissions (in octal) given to all deployed files. For example, if you wanted deployed files to have the permission "-rw-rw-r-x" , then the resulting value would be:

  ```
  file="665"
  ```

- `group` (UNIX only) — specifies the group assigned to all deployed files and directories. This attribute value must be a valid group name or group ID. For example:

  ```
  group="tech_pubs"
  ```
  or
  ```
  group="200"
  ```

  You must also specify the `user` attribute if you use employ the `group` attribute.

- `user` (UNIX only) — specifies the user who will own all deployed files and directories. This attribute value must be a valid user name or user ID. For example:

  ```
  user="jdoe"
  ```
  or
  ```
  user="105"
  ```

  You must also specify the `group` attribute if you use employ the `user` attribute.

- `changeAccess` (Windows only) — modifies the access control lists (ACLs) so that specified users have the designated rights. The new access control entry (ACE) for each specified user allows only the specified rights, discarding any existing ACE. In the following example:

  ```
  changeAccess="{ jdoe:W, tech_pubs:NONE }"
  ```

  any existing ACEs for `jdoe` and `tech_pubs` are removed, `jdoe` is granted write access, and the group `tech_pubs` has no access at all. Any other access rights that may have existed for other users are left unchanged.

- `setAccess` (Windows only) — replaces the ACLs for the deployed files and directories. In the following example:

  ```
  setAccess="{ jdoe:ALL, tech_pubs:RX }"
  ```

  the existing ACL is removed and the user `jdoe` is granted full access. The group `tech_pubs` has read access to the specified files. Any other access rights that may have existed for the file are removed.

Access options specific to UNIX are ignored when deploying to a Windows target host and access options specific to Windows are ignored when deploying to a UNIX target host.

userTranslation

The `userTranslation` element defines information related to changing the user name or user ID while a deployment is in progress. The element maps an existing source user name or ID with a new target user name or ID.

DTD Definition

In the deployment configuration DTD, the `userTranslation` element is defined as:

```
<!ELEMENT userTranslation EMPTY >
<!ATTLIST userTranslation
   from                    CDATA                #REQUIRED
   to                      CDATA                #REQUIRED >
```

Associated Attributes

The following attributes are associated with the `userTranslation` element:

- `from` — specifies the existing source user or user ID (the identification number assigned to each user account with in the UNIX server). For example:

  `from="jdoe"` *or*

  `from="105"`

- `to` — specifies the new target user or user ID. For example:

  `to="rroe"` *or*

  `to="110"`

groupTranslation

The `groupTranslation` element defines information related to changing the group or group ID while a deployment is in progress. The element maps an existing source group or group ID with a new target group name or ID.

DTD Definition

In the deployment configuration DTD, the `groupTranslation` element is defined as:

```
<!ELEMENT userTranslation EMPTY >
<!ATTLIST userTranslation
   from                 CDATA            #REQUIRED
   to                   CDATA            #REQUIRED >
```

Associated Attributes

The following attributes are associated with the `groupTranslation` element:

- `from` — specifies the existing source group or ID (the identification number assigned to each group account with in the UNIX server). For example:

  `from="tech_pubs"` *or*

  `from="100"`

- `to` — specifies the new target group or ID. For example:

  `to="marketing"` *or*

  `to="200"`

deployNRun

The `deployNRun` element is a container for the Deploy and Run feature, which specifies how and when a script is invoked in OpenDeploy.

DTD Definition

In the deployment configuration DTD, the `deployNRun` element is defined as:

```
<!ELEMENT deployNRun ((dnrFile | dnrDir | dnrDeployment)+ ) >
```

Associated Child Elements

The following child elements are associated with the `deployNRun` element:

- `dnrFile` — see page 65.
- `dnrDir` — see page 66.
- `dnrDeployment` — see page 67.

dnrFile

The `dnrFile` element specifies under what conditions deployed files can trigger a Deploy and Run script. Not available for use with transactional deployments.

DTD Definition

In the deployment configuration DTD, the `dnrFile` element is defined as:

```
<!ELEMENT dnrFile (script) >
<!ATTLIST dnrFile
   location          (target)                  #REQUIRED
   when              (before|after)            #REQUIRED
   state             (success|failure|always)  "always"
   mask              CDATA                     #IMPLIED >
```

Associated Child Elements

The following child element is associated with the `dnrFile` element:

- `script` — see page 68.

Associated Attributes

The following attributes are associated with the `dnrFile` element:

- `location` — indicates that the Deploy and Run script is taking place on the target host. There is no default value. You must specify the value `target`.
- `when` — indicates whether the script should be executed before or after the deployment of the particular file. There is no default value. You must specify one of the options.
- `state` — indicates whether the Deploy and Run script should run as a result of the success or failure of the deployment, or whether it should always run in either case. Default value is `always`.
- `mask` — specifies the regular expression specifying the deployed files that will trigger the script. In the following example:

  ```
  mask="\.html$"
  ```

  any deployed file with the file extension `.html` will trigger the Deploy and Run script.

dnrDir

The `dnrDir` element specifies under what conditions deployed directories can trigger a Deploy and Run script. Not available for use with transactional deployments.

DTD Definition

In the deployment configuration DTD, the `dnrDir` element is defined as:

```
<!ELEMENT dnrDir (script) >
<!ATTLIST dnrDir
   location        (target)                    #REQUIRED
   when            (before|after)              #REQUIRED
   state           (success|failure|always)    "always"
   mask            CDATA                       #IMPLIED >
```

Associated Child Elements

The following child element is associated with the `dnrDir` element:

- `script` — see page 68.

Associated Attributes

The following attributes are associated with the `dnrDir` element:

- `location` — indicates that the Deploy and Run script is taking place on the target host. There is no default value. You must specify the value `target`.
- `when` — indicates whether the script should be executed before or after the deployment of the particular directory. There is no default value. You must specify one of the options.
- `state` — indicates whether the Deploy and Run script should run as a result of the success or failure of the deployment, or whether it should always run in either case. Default value is `always`.
- `mask` — specifies the regular expression specifying the deployed directories that will trigger the script. In the following example:

  ```
  mask="cgi-bin$"
  ```

  any deployed directory in the deployment path named `cgi-bin` will trigger the Deploy and Run script.

dnrDeployment

The `dnrDeployment` element specifies under what conditions a deployment can trigger a Deploy and Run script.

DTD Definition

In the deployment configuration DTD, the `dnrDeployment` element is defined as:

```
<!ELEMENT dnrDeployment (script) >
<!ATTLIST dnrDeployment
   location        (source|target)             #REQUIRED
   when            (before|after)              #REQUIRED
   state           (success|failure|always)    "always" >
```

Associated Child Elements

The following child element is associated with the `dnrDeployment` element:

- `script` — see page 68.

Associated Attributes

The following attributes are associated with the `dnrDeployment` element:

- `location` — indicates whether the Deploy and Run script is taking place on the source or target host. There is no default value. You must specify one of the options.
- `when` — indicates whether the Deploy and Run script should be executed before or after the deployment occurs. There is no default value. You must specify one of the options.
- `state` — indicates whether the Deploy and Run script should run as a result of the success or failure of the deployment, or whether it should always run regardless of success or failure. Default value is `always`.

script

The `script` element defines the script associated with the Deploy and Run.

DTD Definition

In the deployment configuration DTD, the `script` element is defined as:

```
<!ELEMENT script EMPTY >
<!ATTLIST script
   cmd              CDATA              #REQUIRED
   as               CDATA              #IMPLIED
   where            CDATA              #IMPLIED
   async            (yes|no)           "no" >
```

Associated Attributes

The following attributes are associated with the `script` element:

- `cmd` —specifies the command where OpenDeploy can start a Deploy and Run script, as well as any accompanying flags or options. You can also specify an executable invocation line. For example:

  `cmd="C:\bin\email_to_admin.bat -user jdoe@interwoven.com"` *or*

  `cmd="/bin/mail jdoe@interwoven.com < /tmp/message.txt"`

  If the command you are going to run requires a scripting engine, the scripting engine must be on the PATH of the user (or system, on Windows) who will be running the script or specified with a full path). For example:

  `cmd="/bin/sh /usr/local/bin/email_to_admin.sh -u jdoe@interwoven.com"`

  or

  `cmd="/usr/local/bin/iwperl /path/to/script.pl"`

- `as` (UNIX only) — specifies a different user name or user ID under which you can run the script. In the following example:

  `as="rroe"` *or*

  `as="110"`

you can run the script as *rroe* rather than as your regular user name. By default, the script runs as the user who invokes OpenDeploy, who will need to be root for most purposes.

- `where` — specifies the path to the location where the `cmd` attribute value is run. For example:

  `where="/tmp"` *or*

  `where="C:\temp"`

  The `where` attribute is optional. If you do not specify a value, the process takes place in the root directory.

- `async` — indicates whether or not to run the script asynchronously. Exercise caution when using this mode, as it could cause many scripts to be run simultaneously. The output from scripts run asynchronously is not captured. Default value is `no`.

logRules

The `logRules` element determines log management features.

DTD Definition

In the deployment configuration DTD, the `logRules` element is defined as:

```
<!ELEMENT logRules >
<!ATTLIST logRules
   maxBytes          CDATA                  #IMPLIED
   level             (verbose|normal)       "verbose" >
```

Associated Attributes

The following attributes are associated with the `logRules` element:

- `maxBytes` — specifies the maximum size in bytes a log file is allowed to grow before the file is closed and OpenDeploy begins writing to a new file. This value is known as the *rollover threshold*. The default `maxBytes` value is 32 megabytes. You can specify different byte measurements in the value, including megabytes (`mb`), kilobytes (`kb`), and bytes (`b`). For example:

  `maxBytes="10mb"` *or*

  `maxBytes="10000kb"` *or*

  `maxBytes="10000000b"`

  indicates that the log file size can grow to 10 megabytes before OpenDeploy will close that log file and start a new one.

  Ensure that you include the proper measurement indicator when setting the threshold size. If no recognizable size measurement is indicated, OpenDeploy uses its default value instead. For example, if the following value was specified:

  `maxBytes="10"`

  OpenDeploy would ignore that stated value and use the default value (`32mb`) instead.

  If the unit of measure is present but unrecognized by OpenDeploy, the default value is used. For example, if the following value was specified:

  `maxBytes="1000x"`

  OpenDeploy would ignore this value and use the default value (`32mb`).

  OpenDeploy will not honor a `maxBytes` value of less than 100 kilobytes (`100kb`). For example, if the following value was specified:

  `maxBytes="50kb"`

  OpenDeploy would ignore this value and use the default value (`32mb`) instead.

- `level` — indicates the level and type of logging OpenDeploy will perform.
  - `verbose` — logs high level of detail on deployment events as they occur. This logging level is best suited for troubleshooting deployment problems or evaluating deployment performance. Verbose logging can create large log files. This is the default logging level.
  - `normal` — logs standard status and error messages. In most cases, this level of logging provides a sufficient amount of detail to meet your needs.

nodeRef

The `nodeRef` element defines the means to access a node referenced within an external configuration file. Nodes can be annotated with a reference to a subsequent deployment and configuration. They can also be annotated as to whether and when that deployment is to be triggered after the successful completion of the current deployment. Note that `targetRules` settings within the `nodeRef` element will override the settings in both the `pathSpecification` and `target` elements.

DTD Definition

In the deployment configuration DTD, the `nodeRef` element is defined as:

```
<!ELEMENT nodeRef (nextDeployment*, targetRules?) >
<!ATTLIST nodeRef
   useNode              IDREF             #REQUIRED >
```

Associated Child Elements

The following child elements are associated with the `nodeRef` element:

- `nextDeployment` — see page 72.
- `targetRules` — see page 52.

Associated Attributes

The following attribute is associated with the `nodeRef` element:

- `useNode` — points to a specific named `node` element attribute previously defined.

nextDeployment

The `nextDeployment` element defines a tiered series of deployments over a range of OpenDeploy servers.

DTD Definition

In the deployment configuration DTD, the `nextDeployment` element is defined as:

```
<!ELEMENT nextDeployment EMPTY>
<!ATTLIST nextDeployment
   deployment              CDATA          #REQUIRED
   invokeOnSuccess         (yes|no)       "no" >
```

Associated Attributes

The following attributes are associated with the `nextDeployment` element:

- `deployment` — specifies the named deployment to execute on the target host upon completion of this current deployment. Used as part of a multi-tiered deployment.
- `invokeOnSuccess` — indicates whether or not to deploy the next tier of deployments based on the success of the first deployment. Default value is `no`.

replicationFarmSet

The `replicationFarmSet` defines one or more `replicationFarm` elements.

DTD Definition

In the deployment configuration DTD, the `replicationFarmSet` element is defined as:

```
<!ELEMENT replicationFarmSet (replicationFarm+) >
```

Associated Child Elements

The following child element is associated with the `replicationFarm` element:

- `replicationFarm` — see page 73.

replicationFarm

The `replicationFarm` element references one or more nodes to be used as targets in a deployment. If more than one node is referenced within a `replicationFarm` it indicates a *fan-out deployment*.

DTD Definition

In the deployment configuration DTD, the `replicationFarm` element is defined as:

```
<!ELEMENT replicationFarm (nodeRef+) >
<!ATTLIST replicationFarm
   name                        ID                #REQUIRED >
```

Associated Child Elements

The following child element is associated with the `replicationFarm` element:

- `nodeRef` — see page 71.

Associated Attributes

The following attribute is associated with the `replicationFarm` element:

- `name` — denotes the unique name of the `replicationFarm` element.

Chapter 4

Command-Line Tools

OpenDeploy command line tools provide functionality by entering the command-line tool and its associated options at the command prompt. In some cases command-line tools provide an alternative to performing tasks in the OpenDeploy user interface. In other cases, a command-line tool is the only method to invoke a particular function or feature.

Command-line tools only can be issued on the host where the OpenDeploy server is installed. Commands can be issued by anyone regardless of whether they hold an Administrator or User role. There are no authentication or authorization checks on individuals invoking command-line tools.

iwodcfg2xml

Converts deployment configurations from the OpenDeploy pre-release 5.0 syntax to the current one for OpenDeploy 5.0 and later. The path to the specified file can be either relative or absolute.

Usage

```
iwodcfg2xml [options] input_file output_file

iwodcfg2xml [options] -dir dir [inExt [outExt]]

iwodcfg2xml [options] -deep dir [inExt [outExt]]
```

`-dir`	Convert all files in the specified directory with a `.cfg` extension. The output files will have the same base and an `.xml` extension. The optional argument `inExt` overrides the default input extension `.cfg`. The optional argument `outExt` overrides the default output extension `.xml`.
`-deep`	Recursively convert all files in *`dir`* and below with a `.cfg` extension. The output files will have the same base and an `.xml` extension. The optional argument `inExt` overrides the default input extension `.cfg`. The optional argument `outExt` overrides the default output extension `.xml`.
`dir`	The name of the directory containing the legacy files.

Options

`-h`	Displays help information.
`-v`	Returns version information.
`-T`	Used to emulate the classic command line argument for a transactional deployment. The default is to generate non-transactional deployments.
`-nodeSetFile` *`file_name`*	This is the file name that will be set as the `nodeSetFile` attribute in the `deploymentConfiguration` element of the output configuration file. The default value is the name of the output node file which is set with the option `-nodesFileOut`.
`-nodesFileIn` *`file_name`*	Sets the name of the file that contains the available nodes. If the file exists and is valid it will be used to supply the node IDs for the targets. The default file name is "odnodes.xml" in the first directory searched.

`-nodesFileOut` *`file_name`*	Sets the name of the file that will be written describing the available nodes. The output file will contain the nodes read from the input file and any new nodes that are needed. If the file exists it will be overwritten and setting this to the node input file effectively updates that file. Any new nodes will be named "*host*:*port*." The default file name is "odnodes.xml" in the first directory searched.
`-masterHost` *`host_name`*	Use *host_name* for the local host name in master (server) configuration. The search order is: 1. This option if supplied. 2. The name of this sever. 3. "Localhost"
`-deployHost` *`host_name`*	Use *host_name* for the local host name in deployment (client) configuration. The search order is: 1. This option if supplied. 2. The "host" statement from the input client .cfg file. 3. Generate an error.
`-onWarning` *`n`*	Set the warning action to *n*. *n*==0: Suppress messages and take no action. *n*==1: Print messages and take no action. (default) *n*==2: Print one message and skip file. *n*==3: Print one message, skip to next file, set exit code to -1. *n*==4: Print one message and exit immediately with -1.

`-onError` *n*	Set the error action to *n*. *n*==0: Suppress messages and take no action. n==1: Print messages and take no action. *n*==2: Print one message and skip file. *n*==3: Print one message, skip to next file, set exit code to -1. (default) *n*==4: Print one message and exit immediately with -1.

Converting Legacy Deployment Configurations Using xml2cfg

Deployment configurations associated with releases prior to OpenDeploy 5.0 are not compatible with this release. However, you can convert your legacy deployment configuration files to a format compatible with this release of OpenDeploy by using the `iwodcfg2xml` command-line tool. The `iwodcfg2xml` command-line tool will convert the legacy deployment configurations to the current syntax, but you might still need to make modifications to ensure total compatibility.

To convert your legacy deployment configurations to the current format, follow these steps:

1. Navigate to the following directory:

   od-home`/bin`

2. Convert your legacy deployment configurations to the current format by entering one of the following commands at the prompt, depending on what kind of conversion you want to perform:

   `iwodcfg2xml` ***`options input_file output_file`*** (single file conversions) *or*

   `iwodcfg2xml` ***`options`*** **`-dir`** ***`dir`*** **`[`*****`inExt`*** **`[`*****`outExt`*****`]]`** (single directory conversions) *or*

   `iwodcfg2xml` ***`options`*** **`-deep`** ***`dir`*** **`[`*****`inExt`*** **`[`*****`outExt`*****`]]`** (nested directory conversions)

   where the following variables apply:

   - ***`options`*** — one or more of the options described later in this section.
   - ***`input_file`*** — the name of the legacy deployment configuration file.
   - ***`output_file`*** — the name of the newly-converted file.
   - ***`dir`*** — the directory where the legacy files reside.

- *inExt* — an alternative file extension to the default `.cfg` for legacy files.
- *outExt* — an alternative file extension to the default `.xml` for converted files.

The `iwodcfg2xml` command-line tool only can be issued on the host where the OpenDeploy base server software is installed. This command can be issued by anyone regardless of whether they hold an Administrator or User role. There are no authentication or authorization checks on individuals issuing this command.

Single File Conversions

You can convert a single legacy deployment configuration to the current format by using the `iwodcfg2xml` command in conjunction with an input file (the legacy file) and an output file (the converted file). The output file must include the `.xml` extension to be run on OpenDeploy.

For example, if you had a legacy deployment configuration `reports.cfg` at the following location:

od-home`/conf/reports.cfg`

you could convert this file by navigating to the following directory:

od-home`/bin`

and entering the following command at the prompt:

`iwodcfg2xml` *od-home*`/conf/reports.cfg` *od-home*`/conf/reports.xml`

Single Directory Conversions

You can convert an entire directory of legacy deployment configurations to the current format using the `iwodcfg2xml` command in conjunction with the `-dir` option and a directory path and name. This command only converts files at the top level of the directory. Files in subdirectories are ignored.

For example, if you wanted to convert all the legacy files in the following directory:

`C:\legacy_files`

you would enter the following command at the prompt:

```
iwodcfg2xml -dir C:\legacy_files
```

The converted files are also contained in the source directory. You cannot redirect converted files to another location when you convert entire directories.

By default, the command will look for legacy files with the extension `.cfg` and convert them to current files with the extension `.xml`. If your legacy files have a different extension, you can note that by adding the input file extension to the command (*`inExt`* in the usage example).

For example, if your directory `C:\legacy_files` contained legacy files with the extension `.txt`, you would enter the following command at the prompt:

```
iwodcfg2xml -dir C:\legacy_files .txt
```

You can also specify a different output file extension by adding the output file extension to the command (*`outExt`* in the usage example). However, converted files must include the `.xml` extension to be usable by OpenDeploy.

Nested Directory Conversions

You can convert an entire directory of legacy files, including those contained in its nested subdirectories, to the current format by using the `iwodcfg2xml` command in conjunction with the `-deep` option and a directory path and name.

For example, if your directory `C:\legacy_files` had the following directory structure:

```
C:\legacy_files
C:\legacy_files\western
C:\legacy_files\eastern\new_york
```

you could convert `C:\legacy_files` and all the legacy files residing in its various subdirectories by entering the following command at the prompt:

```
iwodcfg2xml -deep C:\legacy_files
```

Other than this ability to convert subdirectories, nested directory conversions follow the same rules for file names and extensions as single directory conversion.

iwodpasscoder

Encrypts a string, and returns the encrypted string.

Usage

```
iwodpasscoder -h | -v | string
```

`-h`	Displays usage information.
`-v`	Displays version information.
`string`	String to encode.

iwodschedactivate

Activates or deactivates a scheduled deployment.

Usage

```
iwodschedactivate -h | -v

iwodschedactivate -a deployment -j ID

iwodschedactivate -a "dep_name_pattern" [-j ID]

iwodschedactivate -d deployment -j ID

iwodschedactivate -d "dep_name_pattern" [-j ID]
```

`-h`	Displays usage information.
`-v`	Displays version information.
`-a` *`deployment`*	Activates a specific scheduled deployment.

`-a "dep_name_pattern*"`	Activates a scheduled deployment with an optional *jobID* (`-j` option) using a wild card pattern format. The wild card pattern must be quoted (`"sample*"`). If no `-j` option is present, all scheduled deployments beginning with *dep_name_pattern* will be changed. If a `-j` option is present, only a scheduled deployment beginning with *dep_name_pattern* and having a *jobID* equal to the job identifying number will be changed.
`-d deployment`	Deactivates a specific scheduled deployment, using the *deployment* and `-j ID` options.
`-d "dep_name_pattern*"`	Deactivates a scheduled deployment with an optional job identifying number (`-j` option), using a wild card format. The selection rules are the same as those stated in the schedule activation description above.
deployment	The name of the deployment configuration.
`-j ID`	Specifies a job. The ID number of the deployment. Each time a deployment runs, that deployment is given a unique ID number. Similarly, when you schedule a deployment, that scheduled deployment is also given a issued a unique ID number. Use the `iwodschedget -a` command to see all the ID number for your deployment.

iwodschedadd

Adds a schedule for a deployment configuration.

Usage

```
iwodschedadd -h | -v

iwodschedadd deployment [-r [n][m|h|d|w]] [-s [n][m|h|d|w]]
[-e [n][m|h|d|w]]]
```

`-h`	Displays help information.
`-v`	Displays version information.
`deployment`	Name of the deployment being scheduled.
`-r`	Repeat every *N* minutes, hours, days, or weeks.
`-s [N][m|h|d|w]`	Time from current time to use as start date. The default is 1 minute from current time when the command is entered.
`-e [N][m|h|d|w]`	Amount of time from current time to use as end date. The default end time is none. The scheduled deployment will continue indefinitely.
`n`	A numerical value.
`m`	Minutes.
`h`	Hours.
`d`	Days.
`w`	Weeks.

iwodscheddelete

Deletes an existing scheduled deployment.

Usage

```
iwodscheddelete -h | -v |

iwodscheddelete deployment -j ID

iwodscheddelete "dep_name_pattern*" [-j ID]
```

`-h`	Displays usage information.
`-v`	Displays version information.
`deployment`	The name of the deployment configuration.
`-j` *`ID`*	Specifies a job. The ID number of the deployment. Each time a deployment runs, that deployment is given a unique ID number. Similarly, when you schedule a deployment, that scheduled deployment is also given a issued a unique ID number. Use the `iwodschedget -a` command to see all the ID number for your deployment.
"*`dep_name_pattern`*`*`"	Deletes schedules based on a wild card name selection, with an optional job identifying number (`-j` option). The wild card pattern must be quoted ("*`sample`*`*`"). If the optional job identifying number (`-j` option) is not present, all scheduled deployments beginning with "*`dep_name_pattern`*`*`" will be deleted. If the job identifying number is present, only a scheduled deployment beginning with *`dep_name_pattern`* and having a job identifying number equal to the specified value will be deleted.

iwodschedget

Retrieves scheduled deployments from the Open Deploy scheduler database, and reports selected information back to the user.

Usage

```
iwodschedget -h | -v

iwodschedget -a

iwodschedget -d deployment

iwodschedget -o deployment -j ID
```

`-h`	Displays usage information.
`-v`	Displays version information.
`-a`	Gets all schedules. This is the default option.
`-d` *`deployment`*	Gets all schedules for a particular deployment.
`-o` *`deployment`*	Gets one schedule. Requires the deployment name and the deployment ID number.
`deployment`	The name of the deployment configuration.
`-j` *`ID`*	Specifies a job. The ID number of the deployment. Each time a deployment runs, that deployment is given a unique ID number. Similarly, when you schedule a deployment, that scheduled deployment is also given a issued a unique ID number. Use the `-a` option to see all the ID number for your deployment.

iwodservergetversion

Displays the version of the OpenDeploy server.

Usage

```
iwodservergetversion -h
```

`-h`	Displays help information.

iwodserverreset

Refreshes the OpenDeploy server to the settings specified in the configurations files that have been modified, such as the base server, receiver, and nodes configuration files. The `iwodserverreset` command-line tool will not cause the configuration to be refreshed if there are deployments in progress at the time the command is run.

Usage

```
iwodserverreset -h | -v | -q
```

`-h`	Displays usage information.
`-v`	Displays version information.
`-q`	Disables messages generated when there are active deployments in progress at the time `iwodserverreset` is run.

iwodserverstatus

Displays the status of the OpenDeploy server, including its registry port and the number of active deployments.

Usage

```
iwodserverstatus [-h | -v | -q]
```

`-h`	Displays help information.
`-v`	Displays version information.
`-q`	Omits displaying the number of active deployments.

iwodstart

Starts an OpenDeploy deployment task.

Usage

```
iwodstart -h | -v
```

```
iwodstart deployment [-async] [-inst instance] [-k "key=value"]+ [-sim]
[-V (normal | verbose)]
```

`-h`	Displays usage information.
`-v`	Displays version information.
`deployment`	Name of the deployment to start.
`-async`	Runs `iwodstart` command asynchronously. The `iwodstart` command will return before the deployment completes.
`-inst`	Includes the deployment instance name *`instance`*, which is a suffix that is appended to the deployment name. This option is used to create unique deployment names for each instance of a deployment configuration.

`-k` *`arg`*	Key/value substitution with "*`key=value`*" as the *`arg`* value.
`-sim`	Enables the simulated deployment feature.
`-V` *`arg`*	Logging level with `verbose` or `normal` as args.

The `iwodstart` command returns the following codes regarding the status of the deployment:

- 0 - succeeded
- 1 - starting of the deployment failed
- 2 - deployment ran and returned a failed status

Index

F

G

H

I

J

K

L

M

N